School of Computer Sciences

Semester I, Academic Session 2024/2025

CST435 IoT Architecture and Smart Applications

Assignment 2

Video Demo Link:

https://youtu.be/pxFg3bnB52I

Repository Link:

https://github.com/jake1ee/SmartAgriculture_IoT_Project

Prepared by:

Name	Matric No.	Student Email
Eason Peng	159767	easonpenggg@student.usm.my
Lee Ter Qin	159389	lee0909@student.usm.my

Date of Submission: 24th January 2025

INTRODUCTION

In arid and drought-prone regions, efficient water management is essential for maintaining agricultural productivity. Water scarcity and inefficient irrigation practices often lead to challenges such as overwatering, underwatering, and wastage of valuable resources. To address these issues, our project proposes a Smart Irrigation System that leverages the Internet of Things (IoT) to automate plant watering efficiently.

This system ensures optimal water usage based on real-time soil moisture and weather data, eliminating the need for manual intervention. The proposed system offers a data-driven solution that promotes sustainable water management and innovative agricultural practices by integrating hardware components and cloud platforms. To automate irrigation, the system incorporates multiple sensors, actuators, and a NodeMCU-32S microcontroller. Additionally, the system enables remote monitoring through the V-ONE cloud platform, providing farmers with real-time insights into their irrigation processes.

PROBLEM STATEMENT

Water shortage is a severe issue in many agricultural regions, making efficient irrigation challenging for sustainable farming. According to a report by the World Bank's Water in Agriculture, approximately 70% of global freshwater withdrawals are used for agriculture, with 60% of this water is wasted due to inefficient irrigation practices [1]. Conventional irrigation methods often lack precision, which may lead to unnecessary water usage and high labour costs. Farmers usually rely on subjective evaluations when watering their crops, leading to issues such as the inability to monitor soil moisture levels accurately, excessive water consumption and inefficient water irrigation. These inefficiencies contribute to the loss of water resources, increased operational costs, and reduced crop yields. Farmers can optimize water usage, reduce labour-intensive irrigation practices, and enhance productivity by implementing an automated irrigation system with IoT.

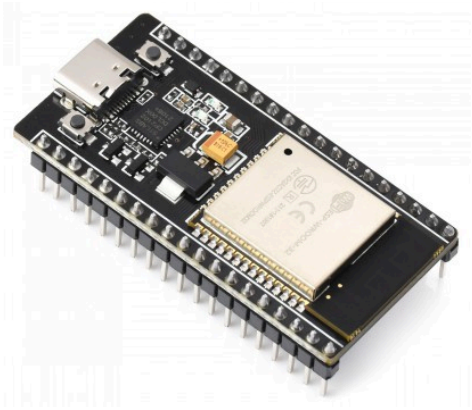
SDG ALIGNMENT

This project aligns with the UN Sustainable Development Goals (SDGs) by addressing issue related to SDG 2: Zero Hunger and SDG 11: Sustainable Cities and Communities. The proposed IoT-based irrigation system contributes to SDG 2 by improving agricultural productivity and ensuring a more reliable food production process. The system encourages sustainable agricultural practices that optimize crop yields while conserving water resources by offering a data-driven approach to irrigation. Additionally, the project also contributes to SDG 11 by improving water resource management and encouraging sustainable urban and rural farming, The IoT system promotes sustainability in agricultural communities by facilitating effective water use and minimizing environmental impact.

OUR IOT SYSTEM

The project will integrate various hardware components, such as microcontrollers, sensors, actuators, and a cloud platform to design a Smart Irrigation System to automate irrigation based on real-time environmental conditions. This system makes real-time irrigation decisions based on predefined thresholds while providing remote cloud access. This leads to a more effective and environmentally friendly irrigation process by enabling farmers to efficiently manage water usage, minimize manual labour, and ensure sustainable agriculture. The hardware and cloud platform that our IoT system uses are described below:

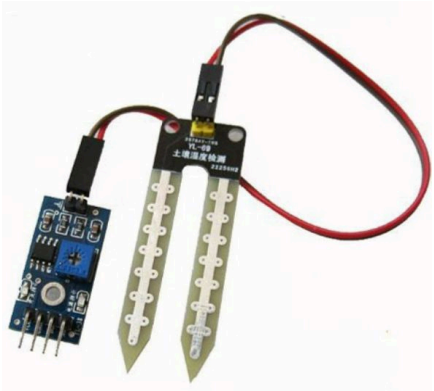
MICROCONTROLLER



A NodeMCU-32S microcontroller is the central processing unit that processes data from the sensors to control the actuators. Additionally, it facilitates communication between the IoT system and the V-ONE cloud platform. The NodeMCU-32S operates on a 5V power supply, supports analog and digital inputs, and is compatible with various sensors and actuators, making it a cost-effective and efficient choice for this project.

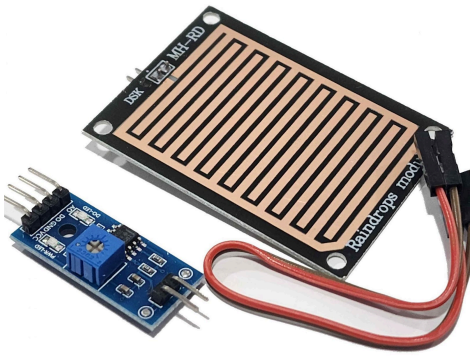
SENSORS

1. **Soil Moisture Sensor**



A soil moisture sensor is placed in agricultural land to monitor the soil moisture content. It provides analog data, which is processed using an LM393 comparator to convert it into a digital signal. The analog data allows for precise detection of soil moisture content, allowing the system to determine whether the rainfall intensity exceeds a predefined threshold. This data collected from the sensor will determine whether irrigation is required.

2. **Rain Drop Sensor**



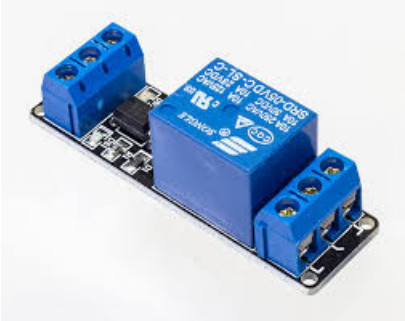
A rain drop sensor is installed in agricultural areas to detect the presence and intensity of rainfall. The sensor provides analog data, which is processed by an LM393 comparator to generate a digital output. This enables the system to determine whether the soil moisture level has dropped below a certain threshold, allowing the conservation of water during wet conditions.

3. **Water Level Sensor**

A water level sensor is placed on the wall of the water supply container to monitor the availability of water. The sensor provides digital data to indicate whether there is sufficient water left in the container. This allows the operation of the water pump to be halted when the water level is insufficient.

ACTUATORS

1. **Optocoupler Relay**



An optocoupler relay serves as a switch to control the water pumps. It activates or deactivates the pumps based on sensor data and predefined conditions.

2. **Water Pump**

A water pump is placed inside the water supply container to deliver inside the container to the plants.

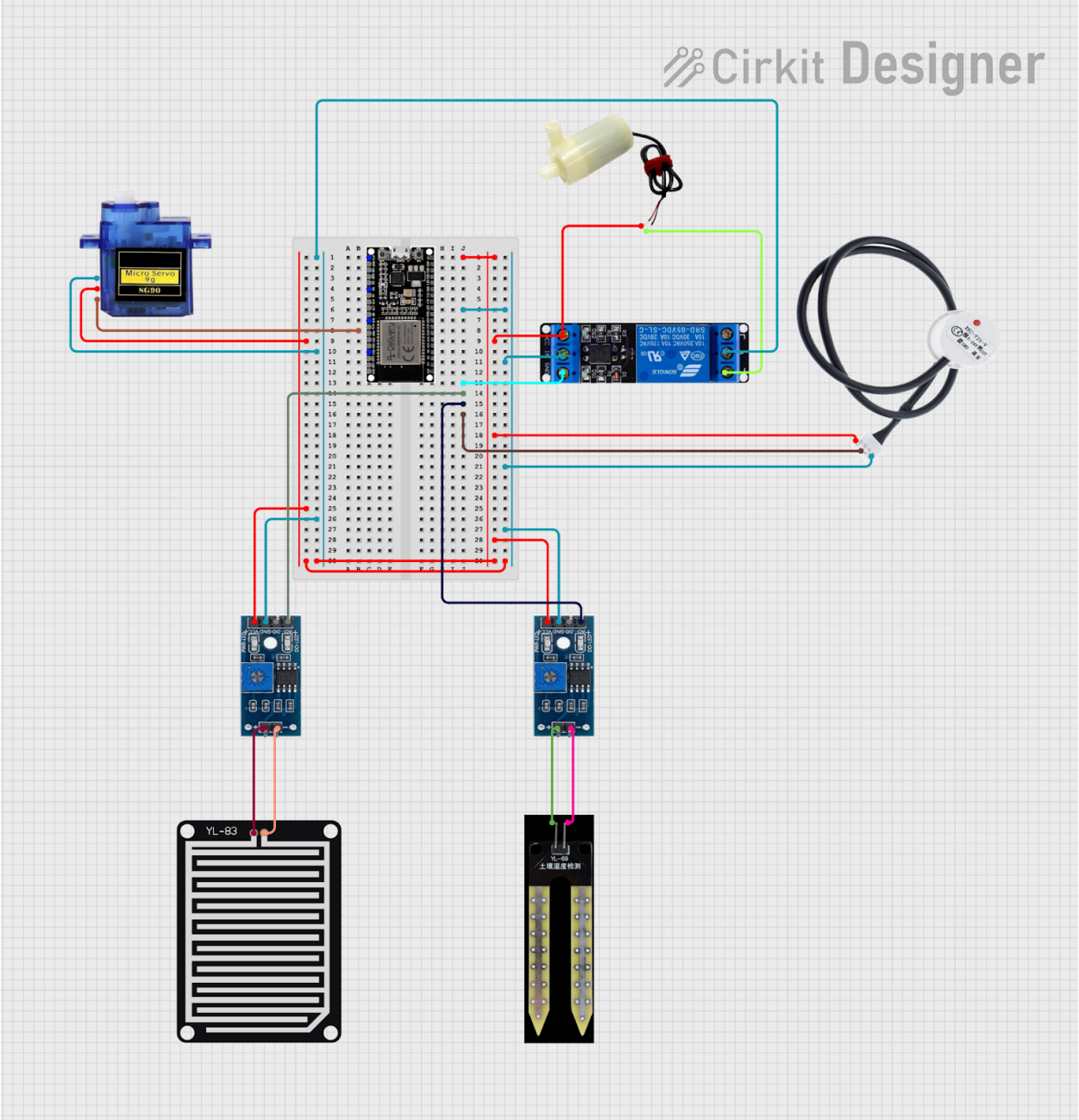
3. **Micro Servo**

A SG90 micro servo is attached to the end of the water pipe of the water pump to rotate the water pipe in a controlled manner to function as a sprinkler, allowing water to cover a wider area.

CLOUD PLATFORM

The V-ONE cloud platform that will be used for providing real-time monitoring and visualization of the data collected from the IoT system. It provides a user-friendly dashboard that allows user to monitor the IoT system remotely to perform data analysis. It has high scalability due to its ability to easily add sensors and actuators into the cloud platform without requiring drastic system redesign.

HARDWARE SETUP



- Breadboard Positive Power Rail (right) -> Breadboard Positive Power Rail (left)
- Breadboard Positive Power Rail (left) -> Breadboard Positive Power Rail (right)
- NodeMCU 5V / Breadboard J1 -> Breadboard Positive Power Rail (right)
- NodeMCU GND / Breadboard J6 -> Breadboard Negative Power Rail (left)
- Optocoupler relay In1 -> NodeMCU Pin 32 / Breadboard J13

- Optocoupler relay DC+ -> Breadboard Positive Power Rail (right)
- Optocoupler relay DC- -> Breadboard Negative Power Rail (right)
- Optocoupler relay COM -> Breadboard Negative Power Rail (right)
- Water pump Anode -> Optocoupler relay DC+
- Water pump Cathode -> Optocoupler relay NO
- Water level sensor OUT -> NodeMCU Pin 39 / Breadboard J16
- Water level sensor VCC -> Breadboard Positive Power Rail (right)
- Water level sensor GND -> Breadboard Negative Power Rail (right)
- SG90 micro servo SIG -> NodeMCU Pin 16 / Breadboard A8
- SG90 micro servo VCC -> Breadboard Positive Power Rail (left)
- SG90 micro servo GND -> Breadboard Negative Power Rail (left)
- LM393 Comparator 1 VCC -> Breadboard Positive Power Rail (right)
- LM393 Comparator 1 GND -> Breadboard Negative Power Rail (right)
- LM393 Comparator 1 AO -> NodeMCU Pin 34 / Breadboard J15
- Soil moisture sensor -> LM393 Comparator 1
- LM393 Comparator 2 VCC -> Breadboard Positive Power Rail (left)
- LM393 Comparator 2 GND -> Breadboard Negative Power Rail (left)
- LM393 Comparator 2 AO -> NodeMCU Pin 35 / Breadboard J14
- Rain drop sensor -> LM393 Comparator 2

SYSTEM FLOW

The IoT system will be connected to the V-ONE cloud platform via the gateway. First, the sensors will be placed in the respective areas. The soil moisture sensor and rain drop sensor will be place at agricultural area, and the water level sensor is place on the outer wall of the water supply container to collect environmental data. The collected data are then be uploaded to V-ONE cloud platform where the data will be collected and display in the dashboard for analysis and visualization. A predefined threshold has been set, so the water pump will only be activated by the relay whenever the conditions are met. The pump stops automatically when soil moisture levels exceed the threshold or if rain is detected or water levels are insufficient. If the water

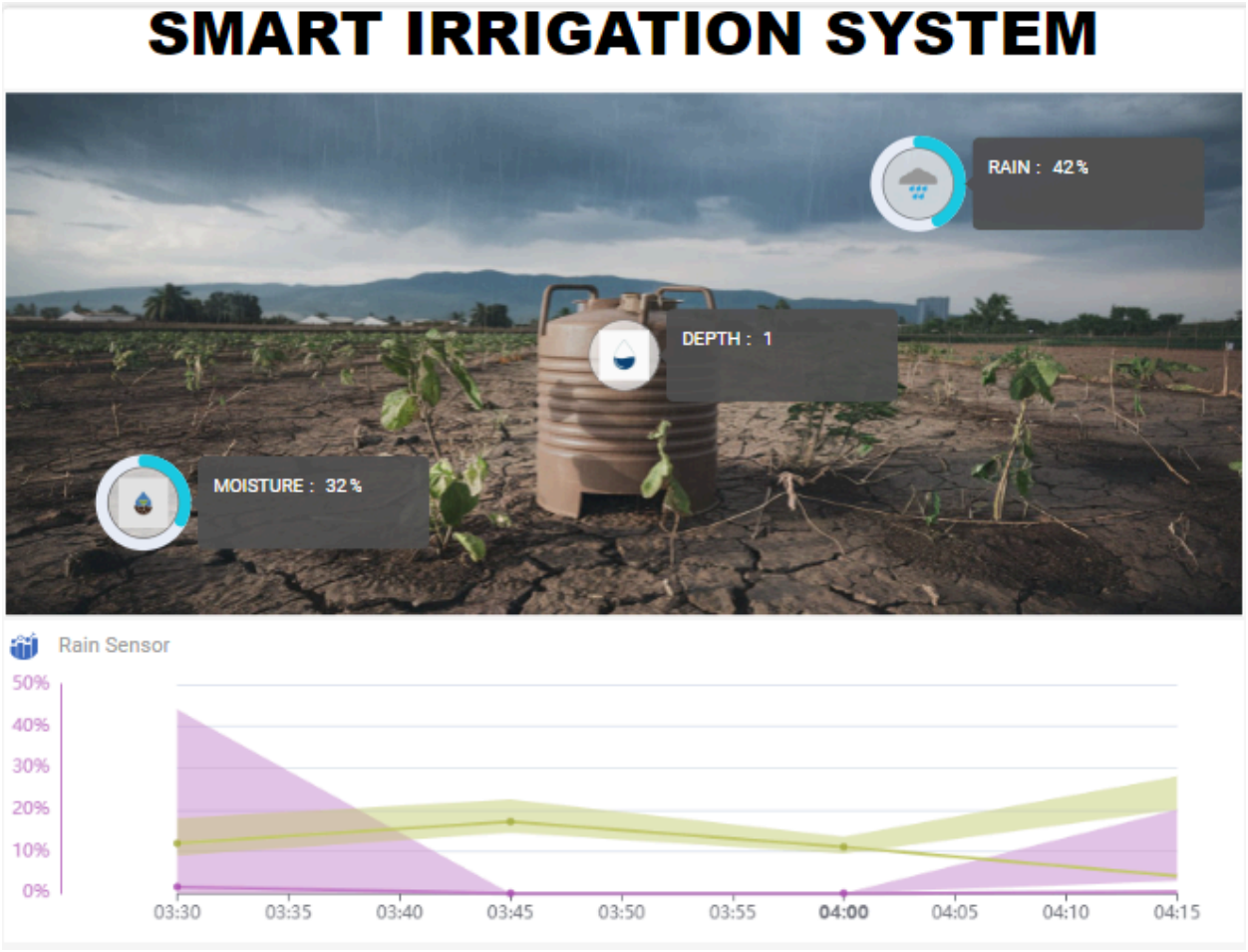
supply in the container is insufficient, the V-ONE cloud platform will send an email alerting the user.

Conditions			Action
Soil Moisture	Rain	Water Level	
≤ 30.0	≤ 40.0	High	Activate water pump
> 30.0	≤ 40.0	High	Deactivate water pump
≤ 30.0	> 40.0	High	Deactivate water pump
> 30.0	> 40.0	High	Deactivate water pump
≤ 30.0	≤ 40.0	Low	Deactivate water pump
> 30.0	≤ 40.0	Low	Deactivate water pump
> 30.0	> 40.0	Low	Deactivate water pump

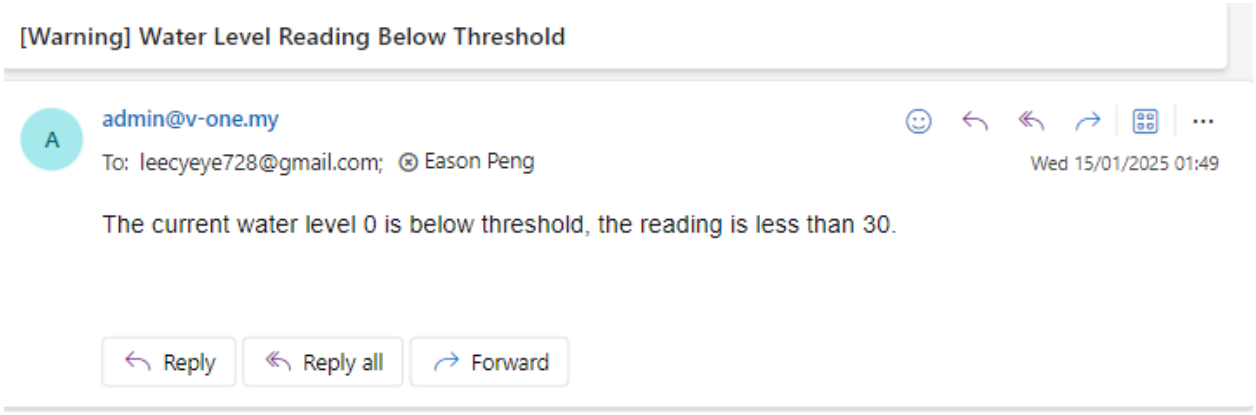
This project provides an innovative and sustainable approach to agricultural water conservation by incorporating IoT technologies into irrigation management. In addition to increasing productivity, irrigation process automation supports global attempts to achieve sustainable farming pratices and water efficiency. Farmers can make well-informed decisions that result in increased crop yields and efficient use of resources by using remote monitoring and real-time data analysis.

DASHBOARD

The V-ONE cloud platform features a user-friendly dashboard that allows farmers to monitor agricultural land conditions remotely. The upper half provides an overview of soil moisture levels, rainfall intensity, and water supply availability. The lower half presents time-series data on soil moisture and weather conditions, enabling farmers to make more informed decisions based on actionable insights.



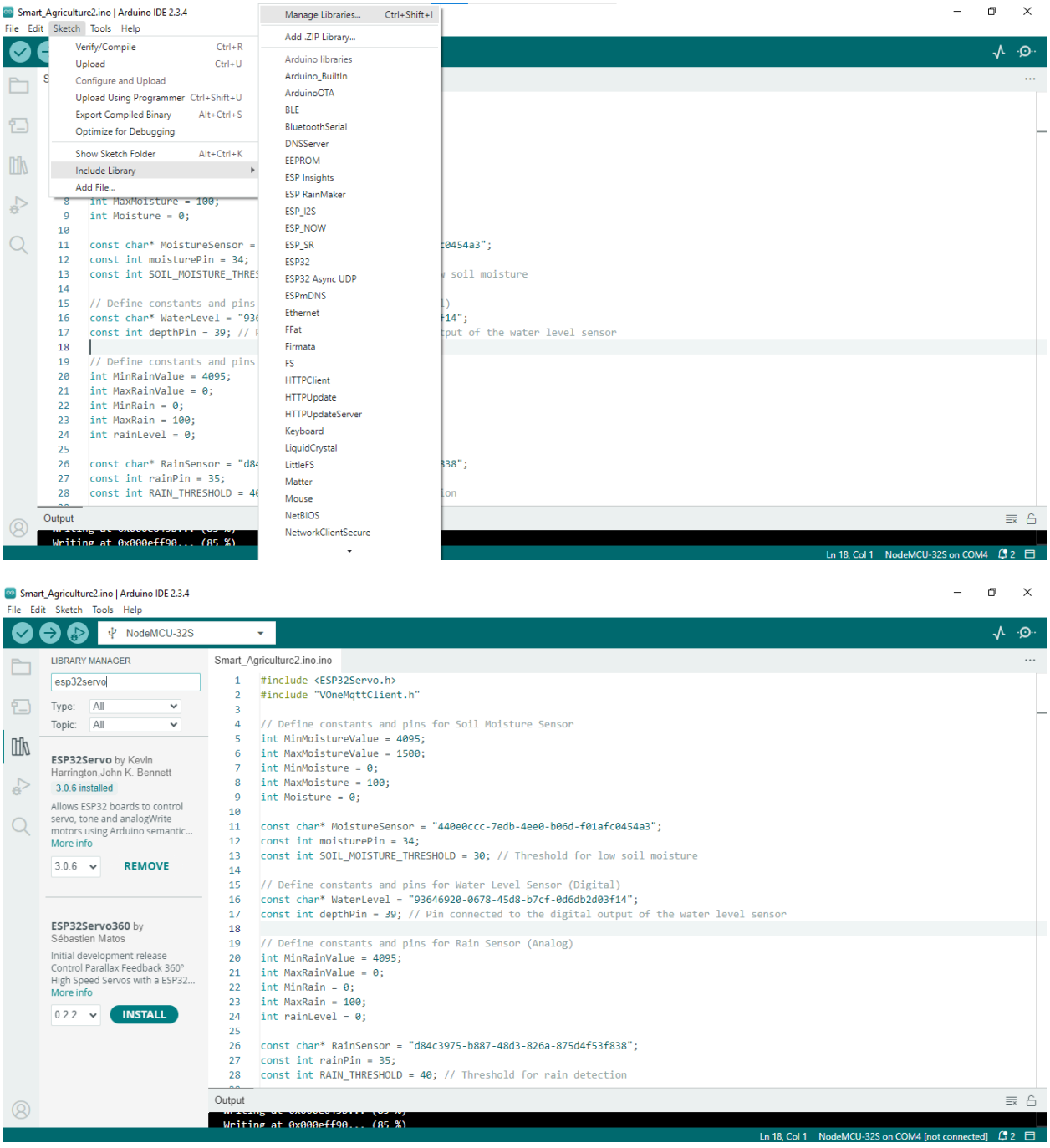
If the water level in the container drops below the recommended height, the user will receive an email alert prompting them to refill the water supply.



CODING

The following explains the smart irrigation system's code developed on the Arduino IDE. When the system first get started, it sets up the WiFi connection and connects to the V-ONE cloud platform via the VOneMqttClient. In order to handle actuator commands from the cloud, it registers a callback function. By default, the servo motor is reset to 0 degrees as the initial starting position. Every sensor pins are configured, and the relay controlling the water pump is set to the off state to prevent unintended activation.

In order to use servo motor on an ESP32 powered board, we must first download the ESP32Servo library. In the top bar of Arduino, go to “Sketch”, choose “Include Library”, then click on “Manage Libraries”. Search for ESP32Servo and install the library.



The system continuously checks the cloud platform's connectivity status within the loop. It publishes the device status and tries to reconnect if the connection is lost. Using the readFilteredAnalog() function, which averages several analog readings for improved accuracy, sensor data is gathered at predetermined intervals. In addition to mapping the rain sensor's value to a percentage, the soil moisture value is converted from raw sensor readings to a percentage

scale (0 to 100). A binary value indicating the presence of water is provided by the digital water level sensor. The cloud then receives these telemetry readings for observation.

A predefined thresholds is used for the water pump's control logic. The water pump is turned on if the water level reach at certain height within the water supply container is detected, there is no heavy rainfall, and the soil moisture level falls below the threshold, else the water pump will stays off. In order to simulate water sprinkler, the servo motor oscillates between 45 and 90 degrees at one-second intervals after the relay is activated. The servo return back to its to initial position when the pump is turned off.

```
#include <ESP32Servo.h>
#include "VOneMqttClient.h"

// Define constants and pins for Soil Moisture Sensor
int MinMoistureValue = 4095;
int MaxMoistureValue = 1500;
int MinMoisture = 0;
int MaxMoisture = 100;
int Moisture = 0;

const char* MoistureSensor = "440e0ccc-7edb-4ee0-b06d-f01afc0454a3";
const int moisturePin = 34;
const int SOIL_MOISTURE_THRESHOLD = 30; // Threshold for low soil moisture

// Define constants and pins for Water Level Sensor (Digital)
const char* WaterLevel = "93646920-0678-45d8-b7cf-0d6db2d03f14";
const int depthPin = 39; // Pin connected to the digital output of the
water level sensor

// Define constants and pins for Rain Sensor (Analog)
int MinRainValue = 4095;
int MaxRainValue = 0;
int MinRain = 0;
int MaxRain = 100;
int rainLevel = 0;

const char* RainSensor = "d84c3975-b887-48d3-826a-875d4f53f838";
```

```
const int rainPin = 35;
const int RAIN_THRESHOLD = 40; // Threshold for rain detection

// Define constants and pins for Relay
const char* Relay = "64a2d9ea-61a9-42f9-888f-f6c170ac6906";
const int relayPin = 32;

// Define constants and pins for Servo
const char* ServoMotor = "5d22ee01-0881-4c4e-9ec6-36d80850d45e";
const int servoPin = 16;
Servo servo;

const int minPulse = 500;
const int maxPulse = 2500;
bool pumpOn = false;

// Create an instance of VOneMqttClient
VOneMqttClient voneClient;

// Last message time
unsigned long lastMsgTime = 0;
const unsigned long LOOP_INTERVAL = 10000; // Adjust the interval as
needed

// Prototype for the callback function
void triggerActuator_callback(const char* actuatorDeviceId, const char*
actuatorCommand);

// Function to control the relay (water pump)
void controlPump(bool turnOn) {
    pumpOn = turnOn;
    if (turnOn) {
        Serial.println("Relay ON: Activating Water Pump");
        digitalWrite(relayPin, HIGH);
    } else {
        Serial.println("Relay OFF: Deactivating Water Pump");
        digitalWrite(relayPin, LOW);
    }
}
```

```
// Filtered analog reading function
int readFilteredAnalog(int pin, int samples = 10) {
    long sum = 0;
    for (int i = 0; i < samples; i++) {
        sum += analogRead(pin);
        delay(10);
    }
    return sum / samples;
}

// Callback function definition
void triggerActuator_callback(const char* actuatorDeviceId, const char*
actuatorCommand) {
    Serial.print("Main received callback : ");
    Serial.print(actuatorDeviceId);
    Serial.print(" : ");
    Serial.println(actuatorCommand);

    if (String(actuatorDeviceId) == Relay) {
        JSONVar commandObjct = JSON.parse(actuatorCommand);
        bool commandValue = (bool)commandObjct["relay"];

        controlPump(commandValue);
                    voneClient.publishActuatorStatusEvent(actuatorDeviceId,
actuatorCommand, true);
    }
}

void setup_wifi() {
    delay(10);
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(WIFI_SSID);
    WiFi.mode(WIFI_STA);
    WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
```

```
    Serial.println();
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
}

// Setup function
void setup() {
    Serial.begin(9600);
    setup_wifi();
    voneClient.setup();
    voneClient.registerActuatorCallback(triggerActuator_callback);

    pinMode(depthPin, INPUT);
    pinMode(rainPin, INPUT);
    pinMode(relayPin, OUTPUT);
    digitalWrite(relayPin, LOW); // Ensure pump is off at startup

    servo.attach(servoPin, minPulse, maxPulse);
    servo.write(90); // Set initial position
}

// Loop function
void loop() {
    if (!voneClient.connected()) {
        voneClient.reconnect();
        voneClient.publishDeviceStatusEvent(RainSensor, true);
        voneClient.publishDeviceStatusEvent(WaterLevel, true);
        voneClient.publishDeviceStatusEvent(MoistureSensor, true);
    }
    voneClient.loop();

    unsigned long cur = millis();
    if (cur - lastMsgTime > LOOP_INTERVAL) {
        lastMsgTime = cur;

        // Soil Moisture Sensor
        int soilValue = readFilteredAnalog(moisturePin);
```

```
            Moisture = map(soilValue, MinMoistureValue, MaxMoistureValue,
MinMoisture, MaxMoisture);
          voneClient.publishTelemetryData(MoistureSensor, "Soil moisture",
Moisture);

        // Water Level Sensor (Digital)
        int waterDetected = digitalRead(depthPin);
                    voneClient.publishTelemetryData(WaterLevel,   "Depth",
waterDetected);

        // Rain Sensor (Analog)
        int rainValue = readFilteredAnalog(rainPin);
          rainLevel = map(rainValue, MinRainValue, MaxRainValue, MinRain,
MaxRain);
        voneClient.publishTelemetryData(RainSensor, "Raining", rainLevel);

        // Control Logic for Water Pump
                if  (Moisture  <  SOIL_MOISTURE_THRESHOLD  &&  rainLevel  <
RAIN_THRESHOLD && waterDetected == HIGH) {
            controlPump(true); // Activate water pump
        } else {
            controlPump(false); // Deactivate water pump
        }
    }

    // Servo Oscillation Logic (Moves left-right continuously when pump is
on)
    if (pumpOn) {
        static unsigned long lastServoMove = 0;
        static bool servoToggle = false;

         if (millis() - lastServoMove > 1000) { // Change direction every
1s
            servoToggle = !servoToggle;
             servo.write(servoToggle ? 45 : 135); // Moves between 45° and
135°
            lastServoMove = millis();
        }
    }
}
```

REFERENCES

[1] World Bank. (2020). Water in Agriculture. Retrieved from World Bank Group Water Global Practice.